

# Might Financial Cryptography Kill Financial Innovation? – The Curious Case of EMV

Ross Anderson, Mike Bond, Omar Choudary,  
Steven J. Murdoch, Frank Stajano

University of Cambridge Computer Laboratory,  
15 JJ Thomson Avenue, Cambridge CB3 0FD, UK  
*name.surname@cl.cam.ac.uk*

**Abstract.** The credit card system has been one of the world’s great successes because of its adaptability. By the mid-1990s, a credit card had become a mechanism for authenticating a transaction by presenting a username (the card number) and a password (the expiry date, plus often a CVV) that was already used in mail order and could be adapted with little fuss to the Internet. Now banks in Europe, and increasingly elsewhere, have moved to the EMV “Chip and PIN” system which uses not just smart cards but also “trusted” hardware. The cryptography supported by this equipment has made some kinds of fraud much rarer – although other kinds have increased, and the jury is still out on the net effect. In the USA in particular, some banks and others oppose EMV on the grounds that it will damage innovation to move to a monolithic and inflexible system.

We discuss the effects that cryptographic lock-down might have on competition and innovation. We predict that EMV will be adapted to use cards as keys; we have found, for example, that the DDA signature can be used by third parties and expect this to be used when customers use a card to retrieve already-purchased goods such as air tickets. This will stop forged credit cards being used to board airplanes.

We also investigate whether EMV can be adapted to move towards a world in which people can use bank cards plus commodity consumer electronics to make and accept payments. Can the EMV payment ecology be made more open and competitive, or will it have to be replaced? We have already seen EMV adapted to the CAP system; this was possible because only one bank, the card issuer, had to change its software. It seems the key to innovation is whether its benefits can be made sufficiently local and incremental. We therefore explore whether EMV can be adapted to peer-to-peer payments by making changes solely to the acquirer systems. Finally, we discuss the broader issue of how cryptographic protocols can be made extensible. How can the protocol designer steer between the Scylla of the competition authorities and the Charybdis of the chosen protocol attack?

## 1 Introduction

The credit card system has supported innovation, both internally and externally, for over half a century. In fact, that's why they succeeded in the first place. During the 1960s and 1970s, they competed with cheque cards that a customer could use to guarantee a cheque to any merchant up to a certain amount. Credit cards won this competition, and a key reason was that they had the flexibility to adapt to mail order and telephone order sales. The card companies found that they'd built not just a settlement system but a global system of authentication where the user name was the card number, and password was the expiry date (joined from the early 1990s by CVVs).

With the arrival of electronic commerce in the mid-1990s, companies such as Microsoft and Netscape tried to design proper cryptographic protocols to support payments (SEPP and STT, amalgamated into SET). However these protocols would have cost time and money to deploy and could not in practice compete with simple credit card transactions which were already available through a deployed infrastructure into which the new e-commerce websites could feed traffic on exactly the same basis as the existing mail-order firms with which they competed. Flexibility won out once more.

There is now a row brewing over the new EMV chip card system developed jointly by Europay, MasterCard and Visa through EMVCo from 1995 onwards. Over the past five years EMV has been deployed in most of Europe and is starting to appear in other countries such as Canada and India. But the USA remains obdurate. US opponents of EMV discuss, inter alia, the question of innovation. Can a much more complex payment system such as EMV adapt to new circumstances and to new market opportunities, or will it fossilise as a platform? Getting hundreds of vendors, thousands of banks and millions of merchants to change their systems simultaneously is extremely hard – that's why it took twenty years to get smart card payments going in the first place!

Yet we have already seen one cycle of innovation. The growth of phishing attacks on electronic banking since 2005 led to the development of the Chip Authentication Program (CAP). This protocol enables a bank to use its issued base of EMV cards to generate codes for two-factor authentication: it issues each customer with a small low-cost reader, and when the customer logs on to the bank website she is asked for an authentication code which she can generate by inserting her card in the CAP reader and typing her PIN. (We described CAP in more detail in [7].) One key fact is that, to introduce CAP, only the card issuing bank had to change anything; no action was required of acquiring banks or of network switches. This causes us to wonder, first, what innovations might be possible to EMV that involve software changes only at a single acquirer. This is typically the bank with which the merchant deposits its card transactions, but acquirers need not be banks; there are established non-bank players like PayPal, and also small startups which provide merchant acquirer services. Second, once we've got to grips with the crypto protocols used in EMV, we'll ask what innovations might be possible with EMV that don't require any banks to change their systems at all.

## 2 Micro-merchant Transactions

Two pervasive problems with payment systems are market concentration and the cost of becoming a merchant. Many countries have only a handful of large banks, and payment services are not very competitive. They suffer from network effects; the current big card brands, of Diners, AmEx, Visa and MasterCard, are what's left of hundreds of payment card schemes, most of which never achieved the necessary critical mass [8]. In the UK, for example, two banks acquire over two-thirds of all credit card transactions from shops; these are the leading banks in the Visa and MasterCard stables respectively. This concentration has harmed innovation. At the beginning of the dotcom boom, small startups who needed credit card acquisition facilities often had to put down large deposits and put up with large merchant discounts – in the range of 5–10% of sales. Non-bank competition, in the form of services such as PayPal and Google Checkout, eventually fixed this problem; these channels accommodate even “micro-merchants” – such as occasional traders who sell goods on eBay.

There are two logical next steps. The first – given that cheques are disappearing throughout Europe, with their abolition in the UK planned for 2017 – is peer-to-peer payments, that is, payments between private individuals. At present, people will write a cheque to a friend or relative to pay for some shopping, or to repay an ad-hoc loan; so what happens after cheques vanish? It would be convenient to be able to wave your bank card over your laptop, or hold it to your mobile phone, in order to make or receive a payment from anyone.

The second (related) step is more competition for cardholder-present transactions. Traditional payment terminals are pricey, and are tied to expensive business banking facilities. So if you sell a handful of goods face-to-face – say at car boot sales, or a church raffle, or perhaps surplus produce from your garden in the summer – cash has been the only real option. Doing a PayPal transaction from my phone to yours is too fiddly!

So an exciting development is the emergence in the USA of services like [squareup.com](http://squareup.com) which will supply you with a tiny credit-card reader to plug into your phone, plus a merchant account to bank your takings. This service enables anyone to become a ‘proper’ merchant and accept credit cards, quickly and cheaply. The service is sufficiently threatening that Verifone is starting to sell similar devices, and the banks are changing standards to require encryption on the link between the reader and the phone – not to block any realistic attack, but to raise the entry costs for other upstart competitors.

So far so good, for micro-merchants in the USA. But what about Canada and Europe? Can we invent a similar service for EMV cards? EMV PIN Entry Devices (PEDs) are a closed market at present. They are supposed to be tamper-resistant, in that it should be hard to bug a merchant terminal to record card and PIN data. But the tamper resistance offered by industry certification schemes has turned out to be illusory [5]; it is be easy to bug a terminal, and many of these devices are compromised every year. Yet even if the certification requirements don't keep out the bad guys, they seem to keep out competition. The PED market is consolidating quickly, and with the recent announcement of a takeover

of Hypercom by Verifone, it looks like we'll be reduced to two major players (the other being Ingenico).

EMVCo is working on contactless payments, where we can either use a credit card with a terminal as before, but with the wire replaced by a wireless channel (typically NFC); or use our mobile phone instead of the card. This won't help the small merchants much if it remains a closed system with certification rules.

In the face of market concentration, poor security, and the lack in the EMV world of a service like [squareup.com](http://squareup.com) to compete at the low end, we need a way to adapt EMV to support cardholder-present low-cost merchant accounts. What are the options?

### 3 Adapting EMV

We are exploring the technical options. Our starting point was that any changes to EMV should require change by either the issuer or the acquirer, but not both. Changing one bank's systems is hard enough; changing 10,000 banks' systems is too much.

#### 3.1 Typical EMV Transaction Flow

First we will describe a typical EMV transaction (this is a much shortened version of the description in [6] to which the reader can refer for the full gory details). As Figure 1 shows, the protocol can be split into three phases: card authentication, cardholder verification and transaction authorization.

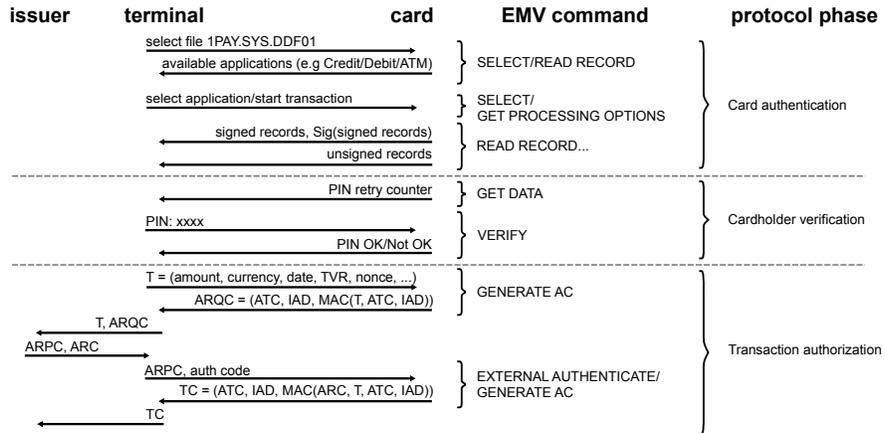


Fig. 1. A complete run of an EMV protocol instance.

**Card Authentication** When a card is inserted into a terminal, it requests a list of supported applications and starts a transaction by sending the **Get Processing Options** command to the card. It next sends **Read Record** commands to read cardholder information including card details (such as primary account number and expiry date), backwards compatibility data (the rest of the magnetic strip), and control parameters (including a list of acceptable cardholder verification methods). There is also an RSA digital signature over a subset of the records, together with a certificate chain to a card scheme root key embedded in the terminal.

In the low-cost variant of EMV, static data authentication (SDA), the card does only symmetric crypto and can only authenticate itself cryptographically to the issuing bank, which must be online for this to work. In the dynamic data authentication (DDA) variant, cards have RSA private keys which are used to sign a 32-bit nonce sent by the terminal.

**Cardholder Verification** The card has a cardholder verification method (CVM) list stipulating when to use a PIN, or a signature, or nothing at all, to authenticate the cardholder, depending on the value of the transaction, its type (e.g. cash, purchase), and the terminal's capabilities. Assuming that the card wishes to verify a PIN, the customer enters it at the terminal which sends it to the card. If it's right, the card returns `0x9000` (this is not cryptographically authenticated by the terminal, but later by the bank from the transaction data).

**Transaction Authorization** In the third step, the terminal asks the card to authenticate the transaction details. The terminal issues the **Generate AC** command to request an ARQC (authorization request cryptogram) from the card. The payload of this command is a description of the transaction, created by concatenating data elements specified by the card in the CDOL 1 (card data object list 1). Typically this includes details like the transaction amount, currency, type, a nonce generated by the terminal, the TVR (terminal verification results) and a sequence number (ATC – application transaction counter). Finally, the card returns the application cryptogram – a message authentication code (MAC), which is calculated over the transaction data with a symmetric key shared between the card and the issuing bank.

The ARQC is then sent by the terminal to the issuing bank, which performs various cryptographic, anti-fraud and financial checks; if it decides to authorise the transaction, it returns a two byte authorization response code (ARC), indicating how the transaction should proceed, and the authorization response cryptogram (ARPC), which is typically a MAC over  $ARQC \oplus ARC$ . These are forwarded by the terminal to the card, which validates them. The terminal asks the card for a transaction certificate (TC) cryptogram, which it sends to the issuer, and also stores in case of dispute. At this point it will typically print a receipt. There are quite a few variants of this transaction flow, such as where the card decides to accept a transaction offline and generates a TC without first

seeing an ARPC; and this complexity has led to some known protocol vulnerabilities [6].

However, although these vulnerabilities mean that the EMV protocol does not entirely keep out the bad guys, so far it has managed to keep out competitors. The system is a closed one, and devices have to be certified tamper-resistant; even although the actual level of tamper-resistance of PIN entry devices is very low [5], the certification system is a closed one, new markets entrants have to sign up to the EMV agreements in order to participate. In this paper, we are interested in how a disruptive competitor might leverage the EMV issued card base and/or infrastructure in order to provide new payment mechanisms without having to get agreement from all the current EMV principals.

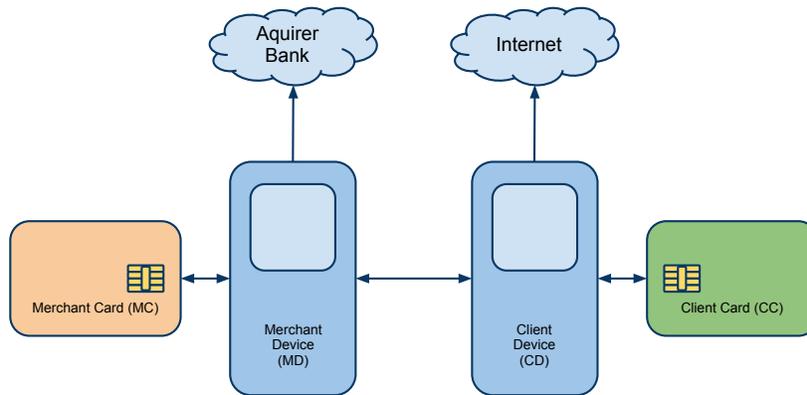
### **3.2 Breaking Tamper-Resistance in Court**

One option might be for a new service provider to go to court to have the tamper-resistance certification standards set aside as a restrictive practice and thus break the Verifone/Ingenico duopoly. The service provider would then supply micromerchants with software for their phone or laptop that implements an EMV terminal (plus, until wireless communication becomes widespread, a cheap smartcard reader). The banks' lawyers would argue that in the absence of trustworthy hardware, malware could harvest card and PIN data; rogue software might also implement the no-PIN attack, so that stolen EMV cards could be used without knowledge of the PIN [6]. The market entrant's lawyers would argue that fraud would be managed in the same way that PayPal or Google Checkout do, by means of fraud-detection systems at the back end. But could we do any better?

### **3.3 Peer-to-Peer EMV – SDA**

Our second possibility is that both customer and merchant have cheap smart card readers, rather than just the merchant. The idea is that the customer will use his own card in his own reader attached to his own phone, and the merchant likewise will use all her own equipment. This largely solves the problems of trusted path and trusted display; malware on the merchant's side can no longer compromise the customer's PIN or display a false payment amount to him. This would be a real improvement on existing systems, whether mag-stripe or EMV: at present, a merchant can program her terminal to display '\$5.00' yet send a transaction for '\$5,000.00' to the card (see Figure 4). There remains the risk of malware on the customer's equipment, which we'll discuss later.

The first variant on this theme is as follows (see Figure 2). Instead of issuing the merchant with a traditional EMV PED, the innovating bank or other acquirer promoting this new system issues her with software for her laptop or mobile phone that assembles the transaction data and submits it to a CAP transaction on her card. The eight-digit CAP code is used as the 32-bit 'unpredictable number' input into an EMV transaction that she sends to the customer. He presents it to his smart card and obtains the ARQC to send to the merchant.



**Fig. 2.** Framework for EMV peer-to-peer.

She relays it in turn to her bank in a quite standard EMV transaction. Her bank verifies the CAP code to check that she is who she claims to be, and if so the main EMV transaction is fed to the switch for onward transmission to the customer's card issuer. From the switch onwards, there is no need to change anything as the EMV transaction flow is unchanged. Table 1 shows the new protocol flow.

The customer's card data can still be hijacked by malware on his own equipment. But if this is the worst that can happen, we have still managed to align incentives rather better than at present: everyone protects their own stuff. There is another minor technical attack in that a crooked merchant might send false transaction data to the customer, who cannot verify the CAP code. This can be fixed in various ways including having the customer send the transaction independently to the acquirer, or notifying the customer by SMS of all transactions. But there are other alternatives.

### 3.4 Peer-to-Peer EMV – Mixed-Mode

EMV cards come in three basic flavours: SDA, DDA and CDA, which are progressively more expensive. An acquiring bank or payment service provider that wants to offer low-cost merchant services can issue its own merchants at least with DDA or CDA cards, regardless of whether local cardholders have them or not. (Most countries use SDA or DDA.)

The merchant (see Figure 2) can now sign the transaction data and provide the first 32 bits of the signature to the customer as the unpredictable number. Things proceed as before, with a standard EMV transaction from the customer via the merchant and the acquiring bank to the customer's card issuer. It does not matter whether this transaction is SDA or DDA, and without loss of generality we can assume the former.

**Table 1.** Protocol flow for a P2P EMV transaction using SDA (see Figure 2 for the entities involved).

<b>dir</b>	<b>data</b>	<b>comment</b>
MD ↔ CD		MD and CD establish a connection via NFC, Bluetooth or other.
CD → MD	CDOL1, CDOL2	Client sends the transaction requirements.
MD → MC	$T = \{\text{Data as per CDOL, with UN}=0\}$	Merchant assembles transaction data as stated by the client's CDOL using Unpredictable Number 0.
MC → MD	$CK = \text{CAP}(T)$	Merchant's card produces 8-digit CAP code.
MD → CD	$T1 = \{T \text{ with UN replaced by CK}\}$	CAP code used as UN for transaction data sent to client.
CD → MD	$AC = \text{MAC}_k(\text{Data}_{CC}, T1)$	Client sends Application Cryptogram (AC) to merchant, which in turn sends this to the acquirer bank.

There is one interesting technical twist. In the DDA protocol, the card signs some data including a 32-bit random nonce provided with the **Internal Authenticate** command, in a protocol designed to let the merchant know that the customer is present. In theory DDA cards can accept longer bit strings depending on the DDOL (Dynamic Data Object List). Tests have indicated that some UK-issued DDA cards accept inputs as large as 1024 bits (128 bytes) regardless of their DDOL. Such cards perform DDA even when we select an application for which DDA is not supported – as stated in the Application Interchange Profile (AIP). However other cards reject any DDA input not matching the DDOL specification. The former behaviour might be caused by incorrect DDA configuration or by design, so we might encounter both types of cards.

Anyway, there's a crypto design problem in respect of those DDA cards that will not sign long strings, as to how to use a device that signs a 32-bit payload to sign a whole transaction. It's no good to just sign the first 32 bits of the hash of the transaction data, as a collision attack can be found trivially. Signing each 32-bit block of a 160-bit SHA-1 hash would be sufficient but involves five transactions. Based on the transaction latency (see table 2) we observe that signing five hash components would take between 2 and 3 seconds, which is less than online authorisations take now.<sup>1</sup> So that's feasible, but we may wish to consider alternatives: the acquirer could issue CDA cards to its cardholders, so

<sup>1</sup> We have managed to obtain 5 consecutive DDA signatures in 2 seconds by omitting the earlier **read data** step in the transaction. Although the specifications say that DDA should be performed after this step, all DDA cards we have tested perform DDA even without reading data first.

**Table 2.** Performance data for different DDA transactions using two cards, one from Visa issued 2009 and another from MasterCard issued 2010. Time for consecutive signature generations includes reset time between transactions. The time measurements have been obtained using an oscilloscope connected to the Smart Card Detective [4], with a 4 MHz clock being provided, and 4 bytes being signed.

Type of transaction	Transaction Time (ms)	
	without application data	reading application data
1 DDA signature Visa	426 (82 used for DDA generation)	1260
1 DDA signature MasterCard	714 (178 used for DDA generation)	1776
5 DDA signatures Visa	2190	6328
5 DDA signatures MasterCard	3610	8936

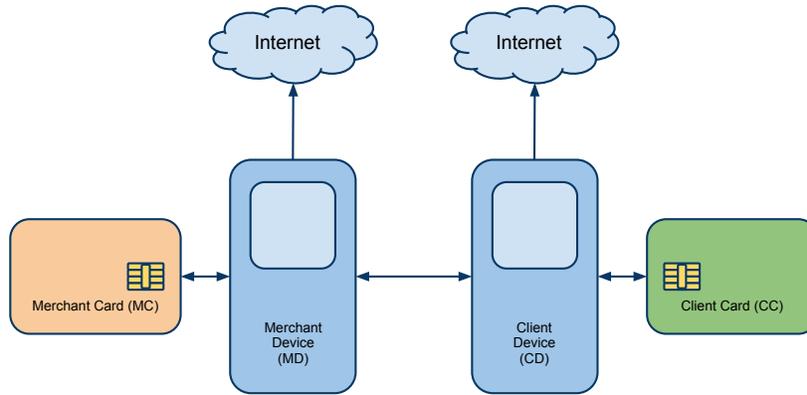
that when acting as merchants they could create full signatures; or the customer and merchant can commit to random numbers and thus jointly select a single block to be signed.

Another potential problem in using the DDA approach was the low availability of the system’s public keys to the customer. DDA was designed for the terminal to verify a string signed by the card, whose key is certified using a closed PKI available to certified terminals only. This is perhaps less of an issue than one might have thought; we’ve been able to find the certificates for Visa cards online. It is also possible to extract root keys from a terminal, as they aren’t as tamper-proof as they’re advertised; another solution would be for individual cardholders and merchants to export their public key certificates to a third party. For example, if you use your RBS MasterCard to top up a PayPal account, you could export the MasterCard public-key certificate to PayPal directly. This opens up other possibilities too.

### 3.5 Going Outside the Banking System

In both the above cases, the banks generally need to consent to the new acquirer’s P2P payment mode, and transfer money between customer accounts when requested. But we must bear in mind the possibility that in order to prevent disruption to their existing revenue streams, some banks will actively block any new modes of operation. Would the new acquirer have to go to court once more, and perhaps in many countries?

A radical alternative is to move the transfer of money out of the hands of the issuers and acquirers, and into the control of a non-bank payment system provider such as Google or PayPal. Customers would associate their cards’ DDA public keys with their payment account, by uploading their public-key certificate and then signing a challenge to prove possession of the card.



**Fig. 3.** Framework for peer-to-peer transactions using EMV for authentication.

When the customers wish to pay for goods, they present their card as normal, but only the initial stage of the EMV transaction occurs, where the card signs one or more terminal-provided nonces derived from transaction data (see above), thus proving to the merchant that the genuine card is present. This digital signature could be sent to the payment service provider, and money transferred. In the case of a customer with a PayPal account that automatically replenishes itself from a conventional credit card account, it might be logical to use that card to authorise PayPal transactions.

Another variant on this theme, which might be slower to take off but which should be less open to legal challenge by the banking industry incumbents, would be for the payment service provider to issue its own cards that would ‘embrace and extend’ – being EMV cards for normal transactions, CDA cards for merchant transactions in the new extended protocol, and also programmed to use their DDA keys to sign transactions in the new system.

In passing, we note that there is an interesting research problem to be tackled here for the DDA cards that only sign 32-bit strings. If the customer and merchant have mobile phones that are capable of public key cryptography, can they get a shared Diffie-Hellman key authenticated using this mechanism? Given that EMV was designed during the ‘Crypto Wars’ – the long tussles during the 1990s over the exportability of cryptographic devices – it may be that DDA signatures were limited to 32 bits specifically to make it harder for people to adapt them for protecting confidentiality. If this was the case, its success is at best marginal. The customer and merchant phones can exchange  $g^x$  and  $g^y$  in the usual way deriving the key  $g^{xy}$  and then authenticate  $h(g^x, g^y, g^{xy})$  by using its first and second 32-bit words as  $N_M$  and  $N_C$  in a signature exchange as set out in table 3. It may be just about possible for a middleperson to set up  $2^{16}$  transactions with different counterparties, and find colliding nonces; but this scarcely gives an industrialisable attack on a real payment system. What’s more, if a couple of

crooks were set on using their bank cards to obtain strong authentication for a home-brew crypto product, they could simply use multiple signatures. We leave as an exercise to the reader the question of whether robust authentication can be obtained without multiple signatures.

**Table 3.** Protocol flow for a P2P transaction using EMV for authentication (see Figure 3 for the entities involved).

<b>dir</b>	<b>data</b>	<b>comment</b>
MD ↔ CD	$g^x, g^y, g^{xy}$	MD and CD establish a connection via Diffie-Hellman
MD → CD	$M_{ID}, N_M$	Merchant sends ID and a nonce $N_M$ derived from the DH data
CD → MD	$C_{ID}, N_C, \text{Signed}_{DDA}(\text{data}_{CC}, N_M)$	Customer sends ID, a DH-derived nonce and DDA signature on $N_M$
MD → CD	$\text{Signed}_{DDA}(\text{data}_{MC}, N_C)$	Merchant sends DDA signature
CD → MD	non-EMV payment	Customer verifies merchant's signature and proceeds with PayPal or other non-EMV payment.

### 3.6 Merchant Authentication

An acquirer or non-bank payment service operator who was implementing a peer-to-peer payment system might want to think long and hard about merchant authentication. If nothing much authenticates the merchant name to the customer, then a micro-merchant might pretend to be a big-name retailer, in a new variant of the phishing attack. So a prudent acquirer who issued CDA cards to merchants might want to include merchant names in card certificates.

The issue that follows on at once from this is naming. An acquirer or payment service operator who certifies the names of payment recipients had better screen them for names similar to established brands, lest it be implicated in a fraud and sued by the brand owner.

Such due diligence is necessary but not sufficient. It is a hard problem to give each of millions of merchants a human-recognisable name that's unique in the universe! Even if well-known corporate brands get some protection from a name screening process, there will still be occasional confusion between small merchants' names, as well as the many disputes where a customer may or may not have bought something from merchant X but gets billed by merchant Y. (The first author recently had to sue his bank in the small claims court to get a refund in such a dispute [2].) So more work may be needed on usability: for example, there may be a case for receipts to contain a globally unique merchant number that can be used to identify the merchant unambiguously at the payment service operator's website.

## 4 Using a Bank Card as a General-Purpose Key

We remarked in the introduction that we would finally look to see if there were any innovative uses for EMV that did not require any bank – issuer or acquirer – to change its systems. Indeed there is one: using an EMV card as a key.

Although a signature on a 32-bit payload may be too small for general-purpose use over a hostile network, it is perfectly acceptable as a means of unlocking or otherwise controlling a device over a trustworthy path. Such a path also deals with the problem of the no-pin attack, in which a middleperson can use a card without knowing the PIN. We will discuss two examples – mobile phones and airline tickets.

If the phone of the future is to have applications that permit one-click payment – whether these are web-hosted apps such as Amazon’s bookstore, or NFC payments that don’t require a PIN – then the security-conscious phone user might use her DDA card as a means of unlocking the phone application that does these. It can also be used to ensure that the right account is selected: she would touch her phone with her NFC Visa card to make a tick payment from the Visa account, and with her NFC debit card to make the payment from her checking account instead. For that matter, a touch with a credit card might be used as a means of unlocking the phone itself, or any other programmable device that can communicate with it.

There is some history behind the idea of using credit cards as keys. A generation ago, hotels tried to use credit cards as room keys, and were blocked when banks objected; undeterred, they started using room keys with the same form factor but proprietary encodings. (And just as mag stripe card vendors developed a secondary market in door locks, so could the smartcard vendors, whose costs of developing DDA cards have been fully amortised by now.)

More recently airlines and rail operators have started asking customers to retrieve pre-booked tickets using the credit card with which they paid for them – a practice to which the banks do not nowadays object. The DDA signature mechanism gives a perfectly good way to implement this; it gives stronger assurance than simply presenting an easily copied static certificate. In this case there is probably not a practical issue with middleperson or collision attacks, as the customer has already done an online transaction that verifies the card number, the billing address and the availability of funds.

There is thus an entirely legitimate use of EMV cards in authenticating air travellers and ensuring that people cannot get on airplanes using forged credit cards. Given the extreme risk-aversion of the aviation security industry, perhaps DDA signatures will be mandated. Should that come to pass, our results show that the implementers will not have to pay rent to the Verifone/Ingenico duopoly. The signature mechanism can be implemented by adding low-cost smartcard readers to airport check-in machines. And given that DDA signature generation is independent of PIN entry, the use of PINs could be omitted or mandated according to the airport’s policy.

## 5 Trustworthy Hardware

The argument may still be made that if a large number of people start to use a peer-to-peer payment mechanism with essentially untrustworthy terminals – commodity smartcard readers connected to mobile phones – then eventually we can expect malware that hijacks customer phones or PCs, stealing card and PIN data. If hotel chains and other businesses start using customers’ bank cards as general-purpose keys, then attacks using dedicated malicious hardware become conceivable too. Of course, research on CAP and on PEDs has shown that no hardware is truly trustworthy; shop terminals are routinely compromised, and a gang might distribute malicious CAPs. That said, dedicated payment terminals can definitely make the attacker’s job harder.

An interesting possible line of development is to combine the concept of the PED (owned by the merchant) and the CAP (owned by the customer) to produce a device that both can use. A hardware device with both a trustworthy display and trustworthy input can simultaneously solve both the display problem and the malware problem. We initially proposed such a device as a solution to the trustworthy display problem in [3], and we have now built a prototype (the Smart Card Detective [4]) which can be seen in Figure 4. A pluggable serial port means that it can be connected to a conventional terminal using a wired smartcard, thus enabling a customer to defend herself against a corrupt merchant or a rogue PED; it could also be connected via USB to a low-cost merchant terminal as described in section 3.1 above; and it could also participate in a peer-to-peer transaction as described in 3.2 above by standing in for the customer’s phone. We used this device to perform the tests reported in this paper.

While this particular design is simply a proof of concept, it illustrates that a low-cost trustworthy smartcard interface, with a display and a control button, is straightforward to design and manufacture.

## 6 Conclusions

Extensibility is the key problem for cryptographers designing real-world protocols and systems. On the one hand, systems such as EMV acquire considerable complexity under their own steam; the vulnerabilities reported in [6], for example, arose because the specification had become unmanageably complex.

Yet innovation will still happen. CAP was one example of how the EMV protocol was adapted to changing business demands. That was possible because the change affected only issuer systems. In this paper we have considered a different change, to adapt EMV to a lower-cost cardholder-present transaction model and ultimately to fully peer-to-peer payments between any two cardholders. This can be done, with the cooperation of some member banks, by changing only acquiring banks’ systems. The lesson from this, we suggest, is to make protocols modular enough that they can be upgraded one side at a time.

We also showed that it was possible to use EMV cards as general-purpose signing oracles. Some cards will sign full-length data while others will only sign



**Fig. 4.** Device with trusted display which can demonstrate to the customer if the protocol is being executed correctly.

32-bit strings; but there are some non-bank applications where such a signature is still very useful, such as authenticating an air passenger by verifying that he possesses the original credit card with which his ticket was purchased. For this reason it seems unlikely that the banking industry will be able to impose monopoly control forever on the uses to which bank cards are put.

As for uses that compete at least in part with the existing bank card system, we described how an acquirer could use the signing facility to support a different, non-bank, payment system. A customer might use her Visa or MasterCard to authenticate transactions on PayPal. The fact that some DDA cards don't do proper signatures is at most a small bump in the road, as there are various workarounds. The biggest limitation may be that DDA signature and PIN verification aren't linked at the protocol level, so in the absence of a trusted path from card to terminal, middleperson attacks may be possible. (But they are anyway on EMV; even if the PED is trustworthy, it doesn't give a trusted path.) Some banks might even fear that fixing this flaw would be against their best interests, as it would make their monopoly easier to break! But the potential interactions between security and competition are complex; at the very least, designers should bear in mind the chosen protocol attack [9].

In earlier work, we discussed the need for better governance within the card systems themselves. EMV has largely escaped from the control of EMVCo; its evolution is being pushed by dozens of vendors, and pulled by thousands of banks. The security mechanisms are not as good as they should be at keeping out attacks, yet they pose real obstacles to constructive innovation. This work showed that the interaction with non-banks, from airline boarding-pass machine vendors through innovative payment startups to the large payment ser-

vice providers, could provide the next wave of disruptive innovation. The forces at play there may pit competition against security; the oligopoly of the brands, banks and terminal vendors (which it's in the public interest to smash) against the chosen protocol attack, which (insofar as it harms cardholders) is in the public interest to avoid. Perhaps the Fed and the European Central Bank will have to step up to the plate and require that the EMV protocol be opened up and brought under proper governance (as suggested in [6]).

### Acknowledgement:

Omar Choudary is a recipient of the Google Europe Fellowship in Mobile Security, and this research is supported in part by this Google Fellowship. However the ideas in this paper were not discussed with any Google staff prior to submission and thus do not necessarily represent the views of Google.

### References

1. Ross Anderson, *'Security Engineering – A Guide to Building Dependable Distributed Systems'*, Wiley 2008
2. Ross Anderson, "How to get money back from a bank", on LightBlueTouchpaper.org (Mar 29 2010), at <http://www.lightbluetouchpaper.org/2010/03/29/how-to-get-money-back-from-a-bank/>
3. Ross Anderson, Mike Bond, "The Man-in-the-Middle Defence", at *Security Protocols Workshop*, Mar 2006 Springer LNCS vol 5087 pp 153–163
4. "The Smart Card Detective: a hand-held EMV interceptor", Omar Choudary, MPhil thesis at University of Cambridge, Computer Lab, <http://www.cl.cam.ac.uk/~osc22/scd/>
5. "Thinking inside the box: system-level failures of tamper proofing", Saar Drimer, Steven Murdoch and Ross Anderson, *IEEE Symposium on Security and Privacy* (2008) pp 281–295; journal version as "Failures of Tamper-Proofing in PIN Entry Devices" in *IEEE Security and Privacy* v 7 no 6 (Nov-Dec 09) pp 39–45
6. "Chip and PIN is Broken", Steven Murdoch, Saar Drimer, Ross Anderson and Mike Bond, at *IEEE Symposium on Security and Privacy* (2010) pp 433–444
7. "Optimised to Fail: Card Readers for Online Banking", Saar Drimer, Steven Murdoch and Ross Anderson, *Financial Cryptography and Data Security 09*, Springer LNCS 5628, pp 184–200
8. DS Evans, R Schmalensee, "Failure to Launch: Critical Mass in Platform Businesses", in *Review of Network Economics* v 9 no 4 (2010), at <http://www.bepress.com/rne/vol9/iss4/1>
9. "Protocol Interactions and the Chosen Protocol Attack", John Kelsey, Bruce Schneier and David Wagner, *Security Protocols – Proceedings of the 5th International Workshop* (1997) Springer LNCS vol 1361 pp 91–104
10. "Verified by Visa and MasterCard SecureCode: or, How Not to Design Authentication", Steven Murdoch and Ross Anderson, *Financial Cryptography 2010*